

Generalised Perceptron Learning

Xiaoyu Wang

XW343@CAM.AC.UK

Department of Applied Mathematics and Theoretical Physics, University of Cambridge

Martin Benning

M.BENNING@QMUL.AC.UK

School of Mathematical Sciences, Queen Mary University of London

Abstract

We present a generalisation of Rosenblatt’s traditional perceptron learning algorithm to the class of proximal activation functions and demonstrate how this generalisation can be interpreted as an incremental gradient method applied to a novel energy function. This novel energy function is based on a generalised Bregman distance, for which the gradient with respect to the weights and biases does not require the differentiation of the activation function. The interpretation as an energy minimisation algorithm paves the way for many new algorithms, of which we explore a novel variant of the iterative soft-thresholding algorithm for the learning of sparse perceptrons.

Keywords: Perceptron, Bregman distance, Rosenblatt’s learning algorithm, Sparsity, ISTA

1. Introduction

In this work, we consider the problem of training perceptrons with (potentially) non-smooth activation functions. Standard training procedures such as subgradient-based methods often have undesired properties such as potentially slow convergence [2, 6, 18]. We revisit the training of an artificial neuron [14] and further show that Rosenblatt’s perceptron learning algorithm can be viewed as a special case of incremental gradient descent method (cf. [3]) with respect to a novel choice of energy. This new interpretation allows us to devise approaches that avoid the computation of sub-differentials with respect to non-smooth activation functions and thus provides better handling for the non-smoothness in the overall minimisation objective during training.

The paper is organised as follows. We begin with a recap of the perceptron model and perceptron learning algorithms in Section 2. Next, we introduce an energy function based on the generalised Bregman distance and demonstrate how Rosenblatt’s learning algorithm can be interpreted as an incremental gradient method applied to this energy in Section 3. In Section 4 we present numerical results for learning sparse perceptrons and compare the results to those obtained from subgradient-based methods. We conclude and give an outlook of future research directions in Section 5.

2. Perceptrons revisited

A (generalised) perceptron can be considered as an artificial neuron [14] or one-layer feed forward neural network of the form

$$y = \sigma \left(W^\top x + b \right). \quad (1)$$

Here σ denotes the (point-wise) activation function, $W \in \mathbb{R}^{m \times n}$ is the weight-matrix and $b \in \mathbb{R}^n$ is the bias-vector. The vector $x \in \mathbb{R}^m$ and the vector $y \in \mathbb{R}^n$ denote the input, respectively the output,

Algorithm 1: Rosenblatt’s Perceptron Learning Algorithm

```

Initialize  $W^0, b^0$ ;
for  $k = 1, 2, \dots$  do
    for  $i = 1, \dots, s$  do
         $e_i = y_i - \sigma((W^k)^\top x_i + b^k)$ 
         $W^{k+1} = W^k + e_i x_i^\top$ 
         $b^{k+1} = b^k + e_i$ 
    end
end

```

of the perceptron. The first perceptron learning algorithm was proposed by Frank Rosenblatt in 1957 [19] and is summarised in Algorithm 1, where s denotes the number of training samples. This early work studies perceptrons in the context of binary supervised classification problems and uses the Heaviside step function as the activation function σ to generate binary output.

Alternatively, the problem of training the weights and biases of a perceptron of the form of Equation (1) can be formulated as a minimisation problem of the form

$$\min_{W, b} F(W, b) := \frac{1}{s} \sum_i^s L(y_i, \sigma(W^\top x_i + b)) + \alpha R(W, b), \quad (2)$$

where F is the overall cost- or loss-function and L is the data function that is chosen a-priori (usually based on prior assumptions of the statistical distribution of the data). The function R is a regularisation function that allows to encode prior information on weight and bias which can be useful to combat ill-conditioning of the data matrix [1] and help to control the validation error [11, 20]. Both terms are balanced with a positive regularisation parameter α .

The overall objective function in Equation (2) is usually minimised by a gradient- or subgradient-based algorithm such as (sub-)gradient descent. If we for instance choose $R \equiv 0$ and $L(y, \sigma(W^\top x + b)) = \frac{1}{2} \|y - \sigma(W^\top x + b)\|^2$ and minimise (2) via mini-batch subgradient descent, we obtain the algorithm described in Algorithm 2, where $B_k \subset \{1, 2, \dots, s\}$ is a (mini-)batch of indices chosen at iteration k . When $|B_k| = 1$, this corresponds to the incremental or stochastic subgradient descent method. Here, σ' denotes the derivative of σ , respectively a subderivative if σ is not differentiable. Hyper-parameters $\tau_w^k > 0$ and $\tau_b^k > 0$ denote the learning rates at iteration k .

We will reveal in the following section that the original Rosenblatt’s learning algorithm is in fact a special case of incremental gradient descent method with respect to a novel choice of energy.

3. Perceptron training: minimising a Bregman loss function

We replace the data term in Equation (2) with a loss function that enables us to interpret Algorithm 1 as an incremental gradient descent method for a special class of activation functions: so-called proximal maps [15].

Definition 1 (Proximal map) *The proximal map $\sigma : \mathbb{R}^n \rightarrow \text{dom}(\Psi) \subset \mathbb{R}^n$ of a proper, lower semi-continuous and convex function $\Psi : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}$ is defined as*

$$\sigma(z) := \arg \min_{u \in \mathbb{R}^n} \left\{ \frac{1}{2} \|u - z\|^2 + \Psi(u) \right\}.$$

Algorithm 2: Mini-batch Subgradient Descent

 Initialize W^0 and b^0 ;

for $k = 1, 2, \dots$ **do**

 Choose $B_k \subset \{1, 2, \dots, s\}$ either at random or deterministically.

$$g_w^k = \frac{1}{|B_k|} \sum_{i \in B_k} [\sigma((W^k)^\top x_i + b^k) - y_i] \sigma'((W^k)^\top x_i + b^k) x_i^\top$$

$$W^{k+1} = W^k - \tau_w^k g_w^k$$

$$g_b^k = \frac{1}{|B_k|} \sum_{i \in B_k} [\sigma((W^k)^\top x_i + b^k) - y_i] \sigma'((W^k)^\top x_i + b^k)$$

$$b^{k+1} = b^k - \tau_b^k g_b^k$$

end

Example 1 (Rectifier) *There are numerous examples of proximal maps, see e.g. [7]. In particular, the rectifier or ramp function can be interpreted as the proximal map of the characteristic function over the non-negative orthant:*

$$\Psi(u) := \begin{cases} 0 & u \in [0, \infty)^n \\ \infty & \text{otherwise} \end{cases} \implies \sigma(z)_j = \max(0, z_j), \forall j \in \{1, \dots, n\},$$

The loss function that we propose in this work is based on a concept known as the (generalised) Bregman distance [4, 5, 9, 13], which is defined as follows.

Definition 2 (Generalised Bregman distance) *The generalised Bregman distance of a proper, lower semi-continuous and convex function Φ is defined as*

$$D_\Phi^{q(v)}(u, v) := \Phi(u) - \Phi(v) - \langle q(v), u - v \rangle.$$

Here, $q(v) \in \partial\Phi(v)$ is a subgradient of Φ at argument $v \in \mathbb{R}^n$ and $\partial\Phi$ denotes the subdifferential of Φ .

Inspired from [10] and based on the definition of the generalised Bregman distance and the assumption $y \in \text{dom}(\Psi)$, we propose the data term

$$L(y, \sigma(z)) := \frac{1}{2} \|y - \sigma(z)\|^2 + D_\Psi^{z - \sigma(z)}(y, \sigma(z)), \quad (3)$$

for the (valid) subgradient $z - \sigma(z) \in \partial\Psi(\sigma(z))$. The motivation in using Equation (3) as a loss function (instead of only using the squared two norm) lies in the simplicity of its gradient.

Theorem 3 *For fixed $y \in \text{dom}(\Psi)$, the gradient of L as defined in Equation (3) with respect to the argument z reads*

$$\nabla_z L(y, \sigma(z)) = \sigma(z) - y.$$

The proof for this theorem is given in the appendix.

If we use the data term defined in Equation (3) in the Minimisation Problem (2) and apply an incremental gradient descent strategy with constant step-size or learning rate one, we immediately obtain Rosenblatt's learning algorithm as defined in Algorithm 1 as a consequence of the chain rule.

Algorithm 3: Rosenblatt-ISTA Algorithm

Initialize W^0 and b^0 ;
for $k = 1, 2, \dots$ **do**
 Choose $B_k \subset \{1, 2, \dots, s\}$ either at random or deterministically.
 $g_w^k = \frac{1}{|B_k|} \sum_{i \in B_k} [\sigma((W^k)^\top x_i + b^k) - y_i] x_i^\top$
 $W^+ = W^k - \tau_w^k g_w^k$
 $W^{k+1} = \arg \min_{W \in \mathbb{R}^{m \times n}} \left\{ \frac{1}{2} \|W^+ - W\|^2 + \alpha \|W\|_1 \right\}$
 $g_b^k = \frac{1}{|B_k|} \sum_{i \in B_k} [\sigma((W^k)^\top x_i + b^k) - y_i]$
 $b^{k+1} = b^k - \tau_b^k g_b^k$
end

In other words, **an incremental or stochastic gradient descent method applied to the Bregman loss (3) generalises Rosenblatt’s original learning algorithm to a broader class of (proximal) activation functions, for which Algorithm 1 can be interpreted as an energy minimisation method**, just with the different Energy (3). We also note that as a special case when the function Ψ is simply zero, the activation function σ is the identity function and (3) is equivalent to the squared loss $\frac{1}{2} \|y - W^\top x\|_2^2$. Hence, incremental gradient descent applied to this energy is equivalent to the Adaline delta rule, first proposed by Bernard Widrow in 1960 [22].

The beauty of interpreting Rosenblatt’s learning algorithm as an energy minimisation problem is that we can apply many other algorithmic strategies to the same energy. In the following section, we demonstrate how we can easily set up an iterative soft-thresholding generalisation of the Rosenblatt algorithm to train a perceptron with sparse weights without having to differentiate the activation function.

4. Application and numerical experiment

In this section, **we use the energy minimisation interpretation to develop a new way of learning sparse perceptrons** (cf. [12]), **without having to differentiate the activation function**. We consider a LASSO-type cost function [21], i.e. we choose the regularisation term R in Equation (2) to be $R(W, b) = \|W\|_1$ in order to promote sparsity of the weight matrix. Here, $\|\cdot\|_1$ denotes the ℓ^1 -norm. We consider using a rectifier activation function as in Example 1. To avoid having to differentiate the non-smooth activation function when solving Minimisation Problem (2), we propose to use the Bregman Data Term (3) as choice for L in combination with the Iterative Soft-Thresholding Algorithm (ISTA) [8] to handle the non-smooth ℓ^1 norm. We name the resulting algorithm **Rosenblatt-ISTA**, which is summarised in Algorithm 3. In numerical experiments, we choose $|B_k| = s$, which corresponds to using the full (sub-)gradient. We compare our approach with three baseline approaches, which aim to solve (2) with a squared Euclidean loss instead. The first approach uses a standard subgradient method with constant step size, where we define the sub-derivative σ' of the rectifier activation function at zero to take on value one. The second approach uses a subgradient method with diminishing step size $\tau_w^k = \frac{\tau_w^0}{\sqrt{k}}$, which guarantees convergence but gives slower convergence. The third method also follows ISTA but instead performs a subgradient descent step in the direction of the (now) non-smooth data term L during each iteration.

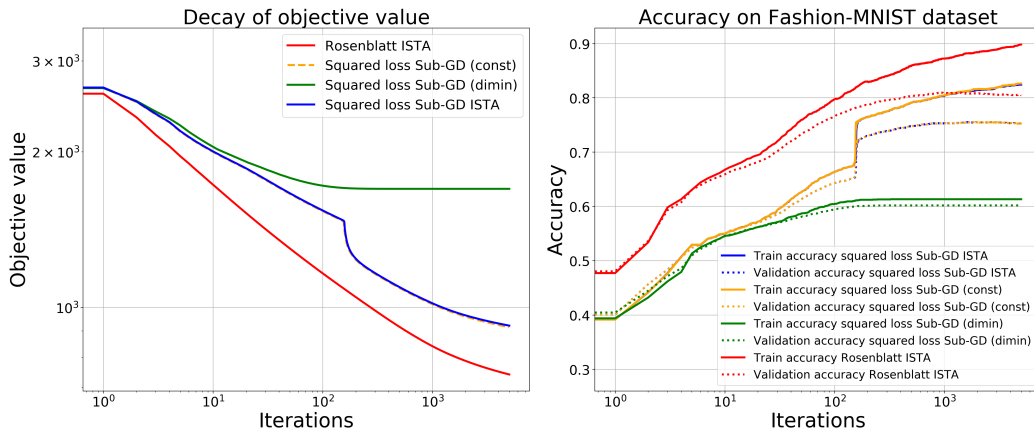


Figure 1: We compare our method of training sparse perceptrons with three baseline approaches as described in Section 4. The plot on the left shows the decay of objective value, i.e. Bregman loss plus ℓ^1 regularisation for our proposed scheme and the squared Euclidean loss plus ℓ^1 regularisation for the baseline approaches. The plot to the right shows the change of training and validation accuracy over the number of iterations. All experiments are performed on the Fashion-MNIST dataset [23].

We test the algorithm performances with a toy classification example based on images from the Fashion-MNIST dataset [23] with the four schemes described earlier. All methods and results are implemented in Python with the Numpy library. We train the perceptrons on 3,000 images selected from the training dataset and validate them on 10,000 images from the test dataset. The regularisation parameters α are set to 0.9 for our proposed objective and 0.81, 0.81, 0.85 for the three baseline methods respectively. The choices of α are determined via cross validation. All four approaches are initialised with the same weight, bias and step size. The overall objective value and accuracy results are visualised in Figure 1. We can see that our proposed training scheme is able to converge faster than the baseline approaches, and also achieves higher training and validation accuracy. It is important that these results are just a proof-of-concept and do not aim at outperforming more complicated neural network architectures.

5. Conclusion and future work

In this work, we discussed the learning of a generalised perceptron model with proximal activation functions. We demonstrated that, from an energy minimisation point of view, the original Rosenblatt’s perceptron learning algorithm can be interpreted as an incremental gradient method with respect to a novel choice of data term, based on a generalised Bregman distance. In addition, this interpretation also generalises the classical Adaline delta rule.

We have modified this interpretation to learn a sparse perceptron with Rosenblatt-ISTA. We showed numerical results for which our approach outperforms three other subgradient-based baseline approaches in terms of achieving both faster convergence and higher accuracy results. A future direction of this work is to extend the proposed generalised perceptron learning scheme to more complex architectures such as multi-layer perceptrons or convolutional neural networks.

References

- [1] Martin Benning and Martin Burger. Modern regularization methods for inverse problems. *Acta Numerica*, 27:1–111, May 2018.
- [2] Dimitri P Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.
- [3] Dimitri P Bertsekas. Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *Optimization for Machine Learning*, 2010(1-38):3, 2011.
- [4] Lev M Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217, 1967.
- [5] Yair Censor and Arnold Lent. An iterative row-action method for interval convex programming. *Journal of Optimization theory and Applications*, 34(3):321–353, 1981.
- [6] Antonin Chambolle and Thomas Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, 2016.
- [7] Patrick L Combettes and Jean-Christophe Pesquet. Deep neural network structures solving variational inequalities. *Set-Valued and Variational Analysis*, pages 1–28, 2020.
- [8] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 57(11):1413–1457, 2004.
- [9] Jonathan Eckstein. Nonlinear proximal point algorithms using bregman functions, with applications to convex programming. *Mathematics of Operations Research*, 18(1):202–226, 1993.
- [10] Jonas Geiping and Michael Moeller. Parametric majorization for data-driven energy minimization methods. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10262–10273, 2019.
- [11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.
- [12] Jeffrey C Jackson and Mark Craven. Learning sparse perceptrons. In *Advances in neural information processing systems*, pages 654–660, 1996.
- [13] Krzysztof C Kiwiel. Free-steering relaxation methods for problems with strictly convex costs and linear constraints. *Mathematics of Operations Research*, 22(2):326–349, 1997.
- [14] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- [15] Jean Jacques Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes Rendus de l’Académie des Sciences de Paris*, A255(22), November 1962.
- [16] Jean-Jacques Moreau. Proximité et dualité dans un espace hilbertien. *Bulletin de la Société mathématique de France*, 93:273–299, 1965.

- [17] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.
- [18] Boris Polyak. *Introduction to Optimization*. Optimization Software, Inc., 1987.
- [19] Frank Rosenblatt. The perceptron, a perceiving and recognizing automaton. *Project Para. Cornell Aeronautical Laboratory*, 1957.
- [20] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [21] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.
- [22] Bernard Widrow. An adaptive 'adaline' neuron using chemical 'memistors', 1553-1552. *Stanford Electronics Laboratories*, 1960.
- [23] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
- [24] Kôsaku Yosida. *Functional analysis*, volume 123. Springer, 1964.

Appendix A. Proof of Theorem 3 in Section 3

Let $E_z(\sigma(z))$ denote the Moreau-Yosida Regularisation [16, 24] of Ψ , where E_z is defined as

$$E_z(x) := \frac{1}{2}\|x - z\|^2 + \Psi(x).$$

We then observe

$$\begin{aligned} L(y, \sigma(z)) &= \frac{1}{2}\|y - \sigma(z)\|^2 + D_{\Psi}^{z - \sigma(z)}(y, \sigma(z)) \\ &= \frac{1}{2}\|y - \sigma(z)\|^2 + \Psi(y) - \Psi(\sigma(z)) - \langle z - \sigma(z), y - \sigma(z) \rangle \\ &= \frac{1}{2}\langle y - \sigma(z), y - \sigma(z) \rangle - \langle z - \sigma(z), y - \sigma(z) \rangle + \Psi(y) - \Psi(\sigma(z)) \\ &= \frac{1}{2}\langle y - z, y - \sigma(z) \rangle + \frac{1}{2}\langle \sigma(z) - z, y - \sigma(z) \rangle + \Psi(y) - \Psi(\sigma(z)) \\ &= \frac{1}{2}\langle y - z, y - z - \sigma(z) + z \rangle - \frac{1}{2}\langle \sigma(z) - z, \sigma(z) - y \rangle + \Psi(y) - \Psi(\sigma(z)) \\ &= \frac{1}{2}\langle y - z, y - z \rangle - \frac{1}{2}\langle \sigma(z) - y, \sigma(z) - z \rangle - \frac{1}{2}\langle y - z, \sigma(z) - z \rangle + \Psi(y) - \Psi(\sigma(z)) \\ &= \frac{1}{2}\|y - z\|^2 - \frac{1}{2}\langle \sigma(z) - y + y - z, \sigma(z) - z \rangle + \Psi(y) - \Psi(\sigma(z)) \\ &= \frac{1}{2}\|y - z\|^2 + \Psi(y) - \frac{1}{2}\|\sigma(z) - z\|^2 - \Psi(\sigma(z)) \\ &= E_z(y) - E_z(\sigma(z)). \end{aligned}$$

It is well known (cf. [17]) that the gradient of the Moreau-Yosida regularisation satisfies

$$\nabla E_z(\sigma(z)) = z - \sigma(z).$$

Therefore, we derive the gradient of L with respect to the argument z as

$$\begin{aligned} \nabla_z L(y, \sigma(z)) &= \nabla_z E_z(y) - \nabla_z E_z(\sigma(z)) \\ &= z - y - z + \sigma(z) \\ &= \sigma(z) - y. \end{aligned}$$

This concludes the proof. □